**Elevens Lab Question Answers**

Activity 2:

*1.* Deck can be composed of many cards. Card can’t be composed of many decks.

*2.* 6

*3.*

{"ace", "2", "3", "4", "5", "6", "7", "8", "9", "10", "jack", "queen", "king"};

{"Spades", "Hearts", "Clubs", "Diamonds"};

{11, 2, 3, 4, 5, 6, 7, 8, 9, 10, 0, 0, 0};

*4.* Yes. ranks and pointValues order must match.

Activity 3:

*1.*

public static String flip()

{

Random gen = new Random();

int num = gen.nextInt(3);

if (num == 0)

{return "heads"; }

else if (num == 1)

{return "heads";}

else

{return "tails";}

}

*2.*

*3.* {3,2,1,4}

Activity 6:

*1.* 5s and 6c, or 6c and 5c

*2.* Must be jack, king, and queen because it would

*3.* I played it so that I would try to not use my jacks, queens, or kings because I thought that was only a last resort because it would open up more possibilities rather than just getting rid of 2 cards

Activity 7:

*1.*

* current cards
* cards in deck
* variable to see if there are no more possible solutions

*2.*

* draw board
* initialize deck
* deal cards
* have user pick cards and swap
* repeat 4th until either wins or loses

*3.* yes

*4a.* in the beginning after everything is initialized

*4b.* anotherPlayIsPossible() and isLegal()

*4c.* 0, 1, 3, 6, 7

*4d.*

public static printCards(ElevensBoard board)

{

List<Integer> cIndexes = board.cardIndexes();

/\* Your code goes here. \*/

for (int i = 0; i < cIndexes.size(); i++)

{

String rank = board.cardAt(cIndexes.get(i).rank());

String suit = board.cardAt(cIndexes.get(i).suit());

int pV = board.cardAt(cIndexes.get(i).pointValue());

System.out.println("Rank: " + rank + ", Suit: " + suit + ", Point Value: " + pV);

}

}

*4e.* anotherPlayisPossible()

Activity 8:

*1.*

State — a deck of cards and the cards “on the” board.

Behavior— to deal the cards, to remove and replace selected cards, to check for a win, to check if selected cards satisfy the rules of the game, to see if there are more legal selections available, and so on.

^^^similarities

differences – they have different ways to check for possibilities and different legal moves, they also have a different number of cards on board and different card values

*2.* using “super()” which calls the super class to initialize everything

*3.* yes they cover the main differences, but differences like different rank values are already covered because they can be initialized differently

Activity 9:

*1.* Because it’s an instance variable that is initialized separately from everything else.

*2.* These tasks are achieved the same way so it is unnecessary.

*3.* Less efficient because you’d have to use more methods that are the same within these three games.